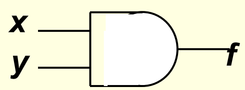
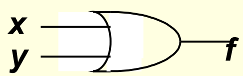
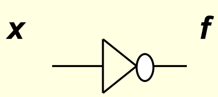
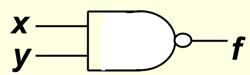
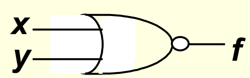
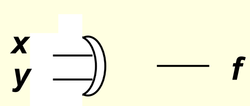
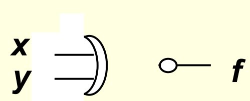
LOGIC GATES AND BOOLEAN ALGEBRA

CHAPTER 2 (CONT…..)

Logic Gates

<table>
<tr><th>Name</th><th>Graphical Symbol</th><th>Algebraic Function</th><th>Truth Table</th></tr>
<tr><td>AND</td><td>
</td><td>$f = x.y$</td><td><table><tr><th>x</th><th>y</th><th>f</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table></td></tr>
<tr><td>OR</td><td>
</td><td>$f = x + y$</td><td><table><tr><th>x</th><th>y</th><th>f</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table></td></tr>
<tr><td>INVERTER</td><td>
</td><td>$f = x'$</td><td><table><tr><th>x</th><th>f</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table></td></tr>
</table>

Logic Gates

Name	Graphical Symbol	Algebraic Function	Truth Table
NAND		***f = (x.y)'***	x y | f 0 0 | 1 0 1 | 1 1 0 | 1 1 1 | 0
NOR		***f = (x + y)'***	x y | f 0 0 | 1 0 1 | 0 1 0 | 0 1 1 | 0
EX-OR		***f = x'y + xy'***	x y | f 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 0
EX-NOR		***f = x'y' + xy***	x y | f 0 0 | 1 0 1 | 0 1 0 | 0 1 1 | 1

Boolean Expression

- Boolean expressions are a much better form for representing digital circuits because it is much easier to manipulate and simplify.
- A Boolean expression is an expression formed with:
  - binary variables
  - the binary operators **OR** and **AND**
  - the operator **NOT**
  - parentheses
  - an equal sign
- For example,
  - $F = \bar{x}y + z$ ***F*** is **1** when *z* = 1 **OR** when *x* = 0 **AND** *y* = 1.
- The precedence of operations is as follows:
  - parentheses,
  - NOT,
  - AND
  - OR.

Boolean Algebra

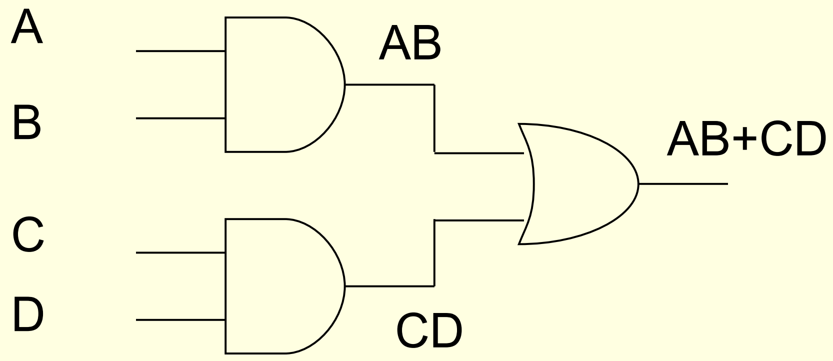
- **Definition**: Theorems that are used at design time to manipulate and simplify Boolean expressions for easier and less expensive implementation.
- Any Boolean expression can be represented using only **AND, OR**, and **NOT** operations.
- May need to use Boolean algebra to change the form of a Boolean expression to better utilize the types of gates provided by the component library being used.
- A Boolean variable, x, can have two values, typically 1 and 0 (on and off)

Laws and Rules Boolean Algebra

Boolean Addition: * $A+0 = A$ * $A+1 = 1$ * $A+A = A$ * $A+\bar{A} = 1$	**Boolean Multiplication:** * $A \cdot 0 = 0$ * $A \cdot 1 = A$ * $A \cdot A = A$ * $A \cdot \bar{A} = 0$
Complement Rules: * $\overline{\overline{A}} = A$	**Boolean other Rules:** * $A+AB = A$ * $A+\bar{A}B = A+B$ $\bar{A}+AB = \bar{A}+B$
Commutative Laws: $A+B = B+A$ $A \cdot B = B \cdot A$	**Associative Laws:** $A+(B+C) = (A+B)+C$ $A(BC) = (AB)C$
Distributive Laws: $A(B+C) = AB+AC$ * $(A+B)(A+C) = A+BC$	Demorgan's Theorem: $\overline{XY} = \overline{X} + \overline{Y}$ $\overline{X+Y} = \overline{X}\,\overline{Y}$

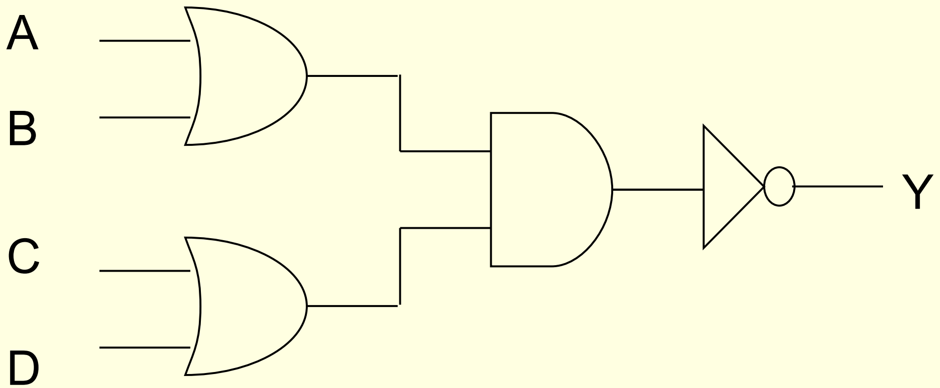
Boolean Analysis of Logic Circuits

- The Boolean Expression for a Logic Circuit
  - Any logic circuit, may completely described using the Boolean expressions.
  - To derive the Boolean Expression of a given logic circuit, begin the left-most inputs and work toward the final output, writing the expression for each gate.
- Ex:

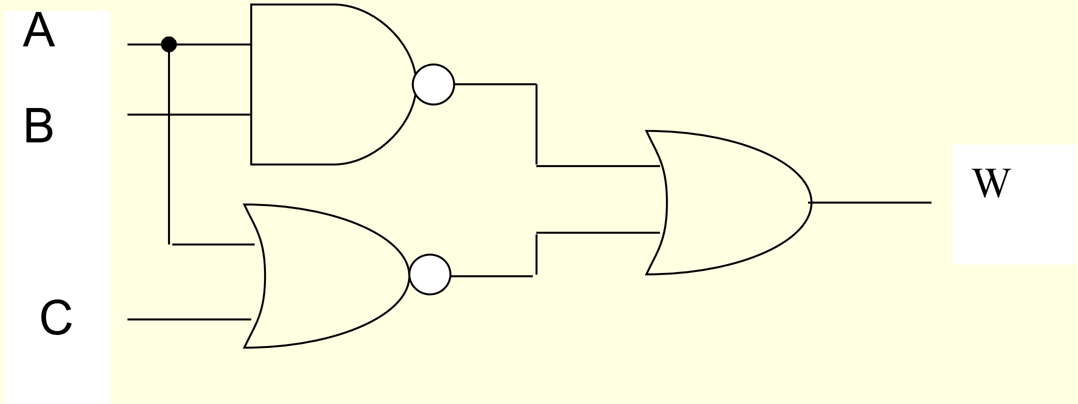


Boolean Analysis of Logic Circuits

- Ex:



- Ex:



Boolean Analysis of Logic Circuits

- Constructing Truth Table for a logic circuit
  - Once the Boolean expression has been determined, truth table can be developed.
  - Boolean expression have to evaluate for all possible combinations of values input variable.
  - If there are four input variables (A, B, C, D) therefore sixteen (2^4=16) combinations of values possible.
- Evaluating the Expression:
  - To evaluate the expression, first find the values of the variables that make the expression equal to 1, using rules for Boolean addition and multiplication.

Boolean Analysis of Logic Circuits

- Putting the Results in Truth Table Format.
  - The first step, list the sixteen input variable combinations of 1s and 0s in binary sequence.
  - Place a 1 in the output column for each combination of input variables.
  - Place a 0 in the output column for all other combinations input variables.

DeMorgan's Theorem

- Apply DeMorgan's theorem to the expression:
  - $\overline{WXYZ}$
  - $\overline{W + X + Y + Z}$
  - $\overline{(A + B + C)D}$
  - $\overline{A\overline{B} + \overline{C}D + EF}$

Simplification Using Boolean Algebra

- Ex: Simplify the Boolean expression
  - $$Y = AB + A(B + C) + B(B + C)$$
  - $$F = (X + Y)(X + \overline{Y})(\overline{X} + Z)$$
  - $$\left[A\overline{B}(C + BD) + \overline{A}\,\overline{B}\right]C$$
  - $$\overline{A}BC + A\overline{B}\,\overline{C} + \overline{A}\,\overline{B}\,\overline{C} + A\overline{B}C + ABC$$
  - $$\overline{AB + AC} + \overline{A}\,\overline{B}C$$
  - $$F = XY + \overline{X}Z + YZ$$

Standard Forms of Boolean Expressions

- All Boolean expressions, regardless of their form can be converted into either of two standard forms:
  - sum-of-product forms
  - product-of-sums forms
- Standardization makes the evaluation, simplification and implementation of Boolean expressions much more systematic and easier.

Standard Forms of Boolean Expressions

- Sum-of-products (SOP) form:
  - Two or more product terms are summed by Boolean addition.
  - OR the outputs of two or more AND gates.
  - Eg: $A\overline{B}C + \overline{A}\,\overline{B} + AB\overline{C}D$
  - A single over bar cannot extend over more than one variable, eg: $\overline{ABC}$
  - Standard SOP-all the variables in the domain apear in each product term, eg: $A\overline{B}CD + \overline{A}\,\overline{B}C\overline{D} + A\overline{B}\,\overline{C}D$

Sum-of-products (SOP)

- Converting product terms to standard SOP
  - Step 1: multiply each nonstandard product term by a term made up of the sum of a missing variable and its complement.
  - Repeat step 1 until all resulting product terms contain all variables.
  - Ex: Convert following Boolean expression into standard SOP form: $A\overline{B}C + \overline{A}\,\overline{B} + AB\overline{C}D$

Standard Forms of Boolean Expressions

- Product-of-Sums (POS) form:
  - Two or more um terms are multiplied by Boolean multiplication.
  - AND the outputs of two or more OR gates.
  - Eg: $(A+\overline{B}+C)(\overline{B}+C+\overline{D})(A+\overline{B}+\overline{C}+D)$
  - A single over bar cannot extend over more than one variable, eg: $\overline{A+B+C}$
  - Standard POS- all the variables in the domain appear in each product term, eg: $(A+B+\overline{C}+D)(A+\overline{B}+\overline{C}+\overline{D})(\overline{A}+B+C+\overline{D})$

Product-of-Sums (POS)

- Converting a sum term to Standard POS
  - Add to each nonstandard product term a term made up of the product of the missing variable and its complement.
  - Repat step 1 untill all esuling sum terms contain all variables.
  - Ex: Convert the following Boolean expression into standard POS form: $\left(A+\overline{B}+C\right)\left(\overline{B}+C+\overline{D}\right)\left(A+\overline{B}+\overline{C}+D\right)$

Standard Forms of Boolean Expressions

- Converting Standard SOP to Standard POS
  - Step1: Determine the binary numbers for each literal.
  - Step2: Determine the absent binary numbers.
  - Step3: Write the binary numbers from Step 2 in equivalent term in opposite standard

- Ex: Convert the following SOP expression to an equivalent POS expression

$$\overline{A}\,\overline{B}\,\overline{C} + \overline{A}B\overline{C} + \overline{A}BC + A\overline{B}C + ABC$$

Converting Standard SOP & Standard POS to Truth Table Format

- SOP- equal to 1 only if at least one of the product terms is equal to 1.
- POS-equal to 0 only if at least one of the sum terms is equal to 0.

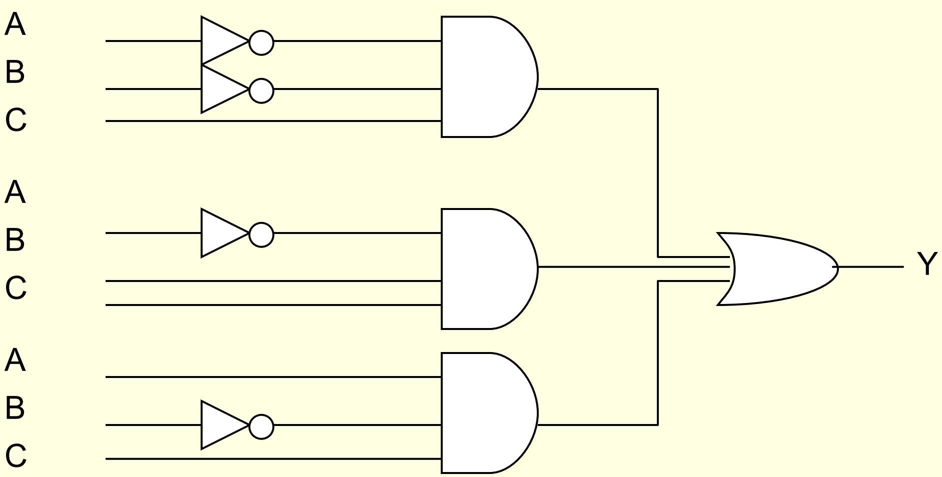
Converting Standard SOP to Truth Table Format

A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

$\overline{A}\,\overline{B}C$

$\overline{A}BC$

$A\overline{B}C$



- Ex: Develop the truth table for the expression $Y = \overline{A}BC + AB\overline{C}$ and draw the logic circuit.

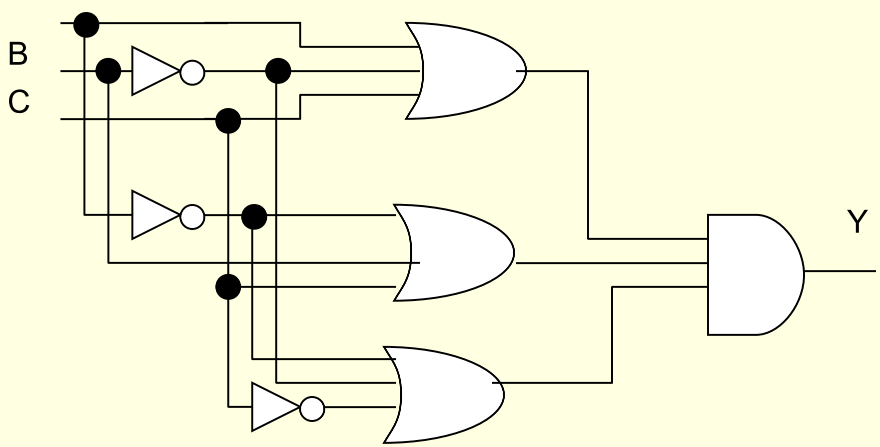
Converting Standard POS to Truth Table Format

A	B	C	Y
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

$\rightarrow A+\overline{B}+C$

$\rightarrow \overline{A}+B+C$

$\rightarrow \overline{A}+\overline{B}+\overline{C}$



- Ex: Develop the truth table for $\left(A+\overline{B}\right)\bullet\left(\overline{A}+B\right)\bullet\left(A+B\right)$ and draw logic circuit.

The Karnaugh Map (K-map)

- Is an array of cells in which each cell represents a binary value of the input variables.
- Can be used for expressing two, three, four, and five variables.
- Mapping a nonstandard SOP expression:
  - A Boolean expression must first be in standard form before putting in Karnaugh map.

The K-Map SOP Minimization

- Grouping/looping the 1s
  - Each cell in a group must be adjacent, this included wrap-around adjacent.
  - Maximize the size of the groups, but minimize the number of groups.
  - A group must contain either 1,2,4,16 cells (power of 2).
  - Overlapping groups is allowed.
- Determine the minimum SOP expression
  - Each group of cells containing 1s creates one product terms composed of all variables that occur either uncomplemented or complemented.
  - Variables with both uncompleted and complemented within the group are eliminated.

2 variables K-map

- A two variable K-map with their truth table which is showing product terms of each line as shown below

A	B	Product term
0	0	$\overline{A}\,\overline{B}$
0	1	$\overline{A}B$
1	0	$A\overline{B}$
1	1	AB

	B	B
Ā	$\overline{A}\,\overline{B}$	$\overline{A}B$
A	$A\overline{B}$	AB

- Ex: Map the following SOP expression of two variables. $\overline{AB} + A\overline{B} + AB$

2 variables K-map

A	B	X
0	0	$1 \rightarrow \bar{A}\bar{B}$
0	1	0
1	0	0
1	1	$1 \rightarrow AB$

$$\left\{ X = \bar{A}\bar{B} + AB \right\}$$

	$\bar{B}$	B
$\bar{A}$	1	0
A	0	1

(a)

3 Variable K-map

A	B	C	Product terms
0	0	0	$\overline{A}\,\overline{B}\,\overline{C}$
0	0	1	$\overline{A}\,\overline{B}C$
0	1	0	$\overline{A}B\overline{C}$
0	1	1	$\overline{A}BC$
1	0	0	$A\overline{B}\,\overline{C}$
1	0	1	$A\overline{B}C$
1	1	0	$AB\overline{C}$
1	1	1	ABC

	$\overline{C}$	C
$\overline{A}\,\overline{B}$	$\overline{A}\,\overline{B}\,\overline{C}$	$\overline{A}\,\overline{B}C$
$\overline{A}B$	$\overline{A}B\overline{C}$	$\overline{A}BC$
$A\overline{B}$	$A\overline{B}\,\overline{C}$	$A\overline{B}C$
AB	$AB\overline{C}$	ABC

- Ex: Map the SOP expression $\overline{A}\,\overline{B}C + \overline{A}B\overline{C} + \overline{A}BC + A\overline{B}C + ABC$ of three variables.

3 Variable K-map

A	B	C	X
0	0	0	$1 \rightarrow \bar{A}\bar{B}\bar{C}$
0	0	1	$1 \rightarrow \bar{A}\bar{B}C$
0	1	0	$1 \rightarrow \bar{A}B\bar{C}$
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	$1 \rightarrow AB\bar{C}$
1	1	1	0

$$\left\{ \begin{aligned} X = {} & \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C \\ & + \bar{A}B\bar{C} + AB\bar{C} \end{aligned} \right\}$$

	$\bar{C}$	C
$\bar{A}\bar{B}$	1	1
$\bar{A}B$	1	0
AB	1	0
$A\bar{B}$	0	0

(b)

4 Variable K-map

A	B	C	D	X	
0	0	0	0	0	
0	0	0	1	1	$\rightarrow \bar{A}\bar{B}\bar{C}D$
0	0	1	0	0	
0	0	1	1	0	
0	1	0	0	0	
0	1	0	1	1	$\rightarrow \bar{A}B\bar{C}D$
0	1	1	0	0	
0	1	1	1	0	
1	0	0	0	0	
1	0	0	1	0	
1	0	1	0	0	
1	0	1	1	0	
1	1	0	0	0	
1	1	0	1	1	$\rightarrow AB\bar{C}D$
1	1	1	0	0	
1	1	1	1	1	$\rightarrow ABCD$

$$\left\{ \begin{array}{l} X = \bar{A}\bar{B}\bar{C}D + \bar{A}B\bar{C}D \\ \quad + AB\bar{C}D + ABCD \end{array} \right\}$$

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	1	0	0
$\bar{A}B$	0	1	0	0
AB	0	1	1	0
$A\bar{B}$	0	0	0	0

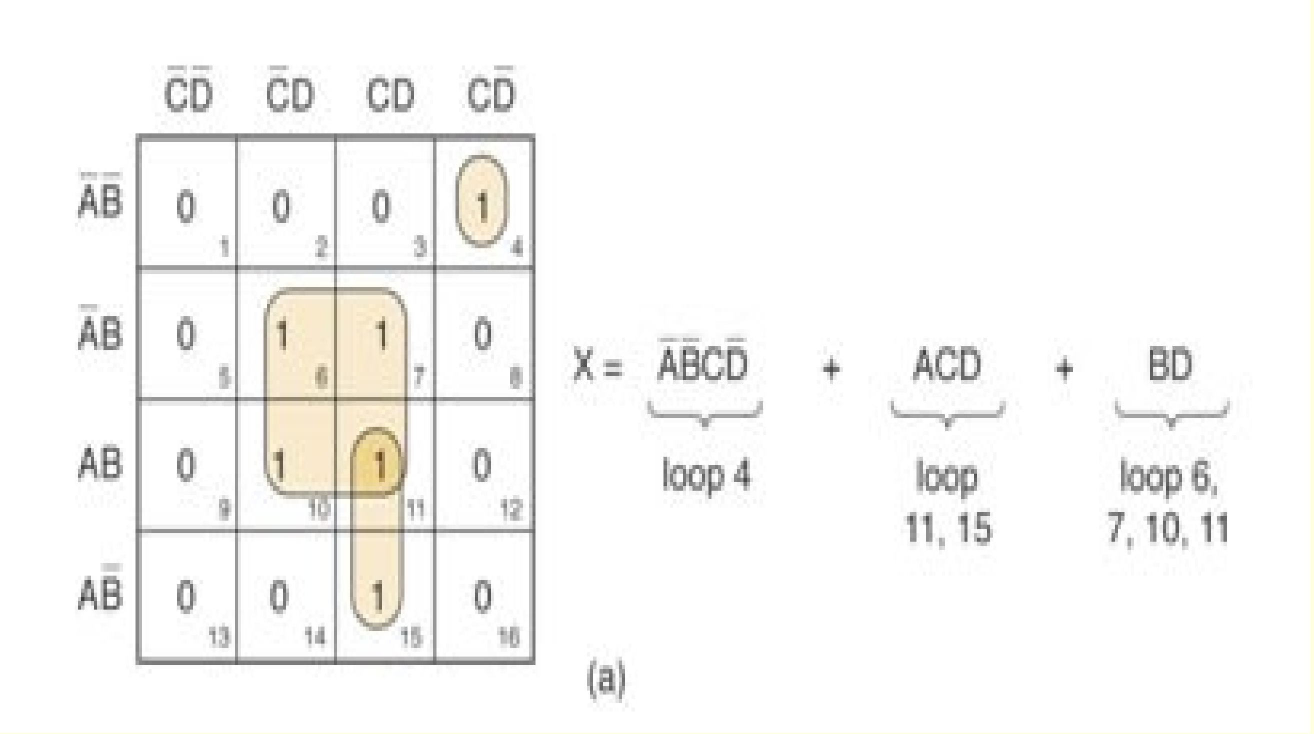
(c)

4 Variable K-map

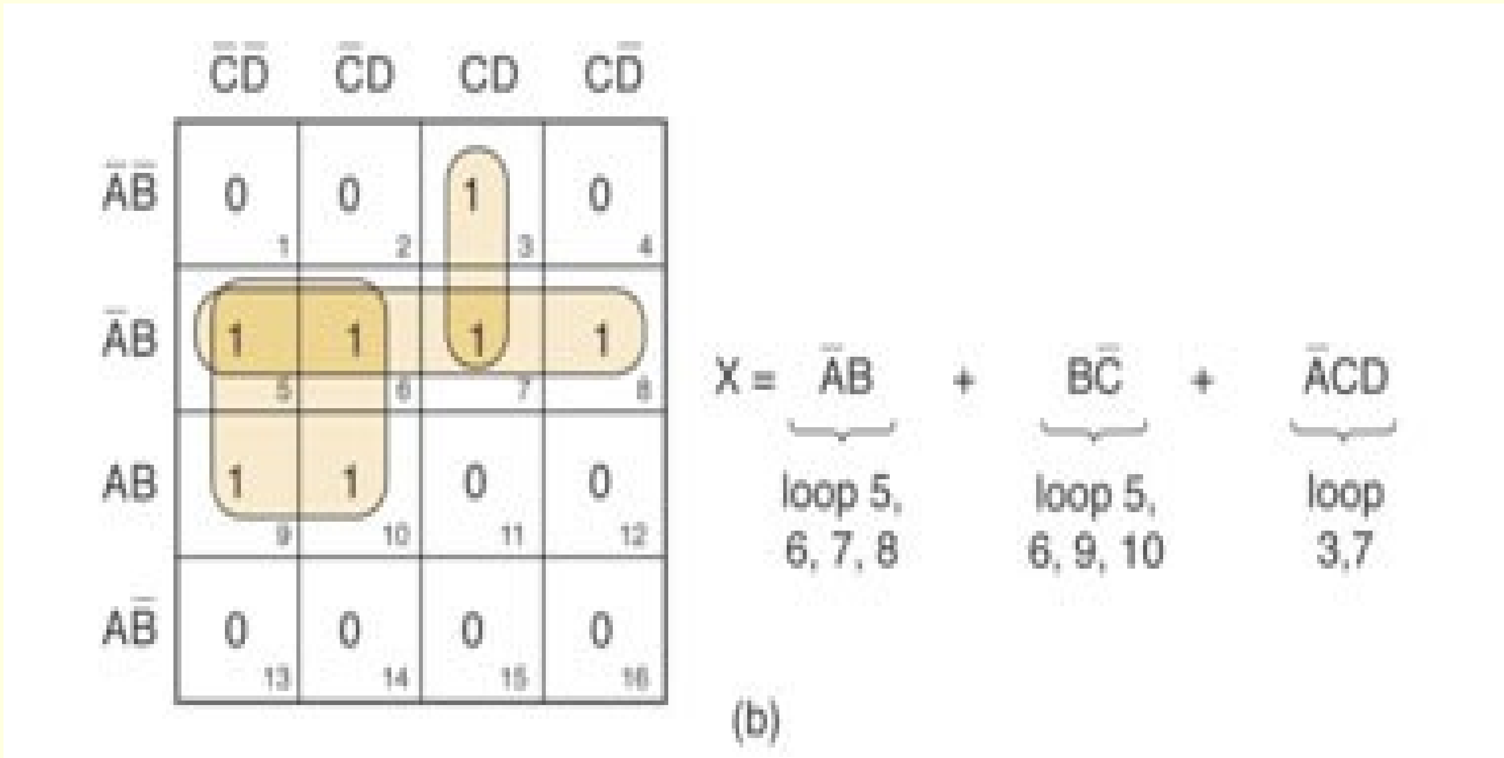
- Ex: Map the following four variable SOP expressions:

$$\overline{A}B\overline{C}\,\overline{D} + A\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}B\overline{C}D + AB\overline{C}D + A\overline{B}\,\overline{C}D + \overline{A}BCD + A\overline{B}CD + ABCD$$

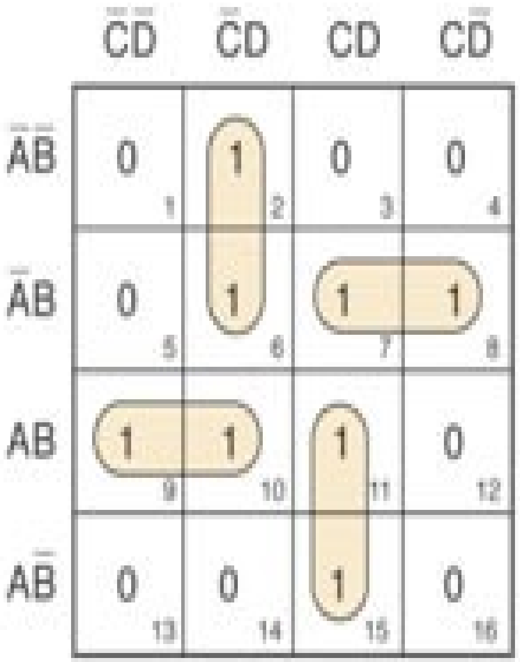
More examples



More examples



More examples



$$X = \underbrace{AB\bar{C}}_{9,\ 10} + \underbrace{\bar{A}\bar{C}D}_{2,\ 6} + \underbrace{\bar{A}BC}_{7,\ 8} + \underbrace{ACD}_{11,\ 15}$$

(c)

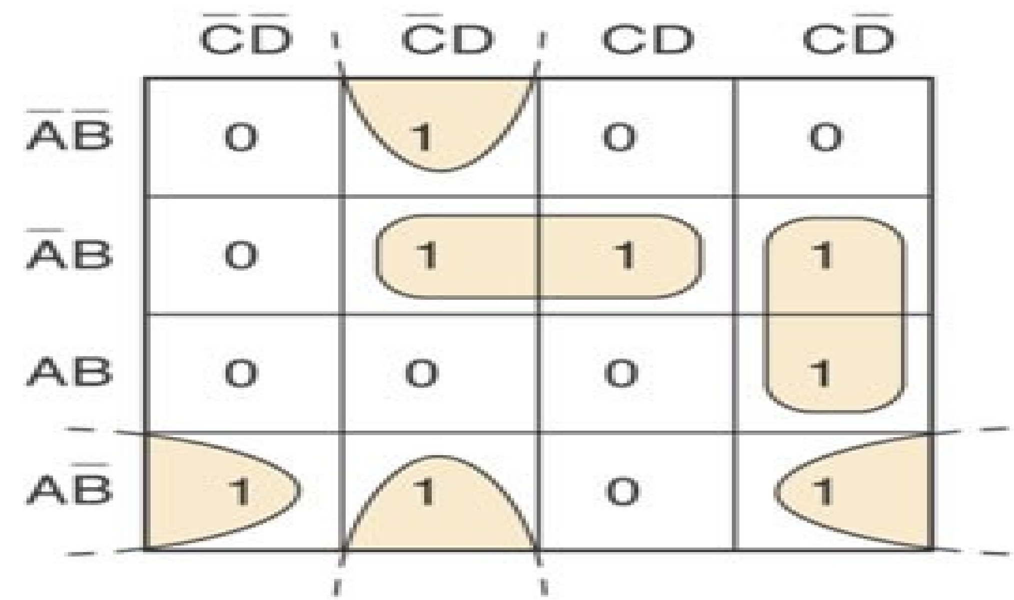
More examples

	$\overline{C}\,\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$
$\overline{A}\,\overline{B}$	0	1	0	0
$\overline{A}B$	0	1	1	1
AB	0	0	0	1
$A\overline{B}$	1	1	0	1

$$X = \overline{A}\,\overline{C}D + \overline{A}BC + A\overline{B}\,\overline{C} + AC\overline{D}$$

(a)

More examples



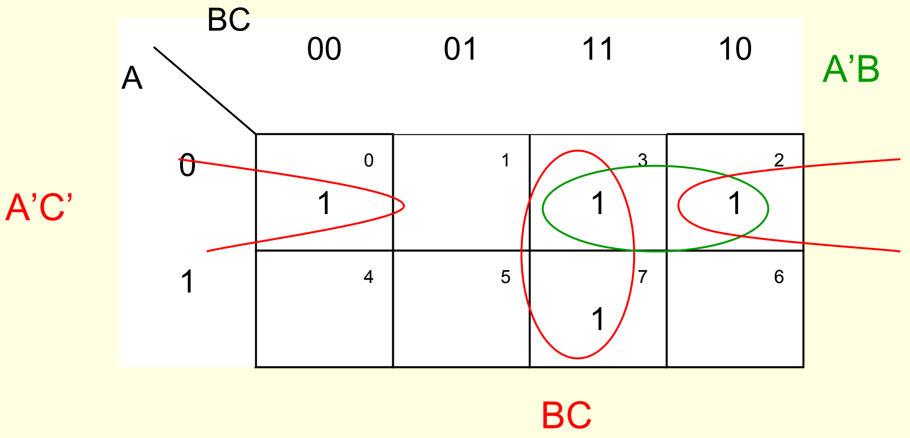
$$X = \bar{A}BD + BC\bar{D} + \bar{B}\bar{C}D + A\bar{B}\bar{D}$$

(b)

More examples

- Use a K-map to simplify the following Boolean function:

$$F(A,B,C) = \Sigma m(0,2,3,7)$$

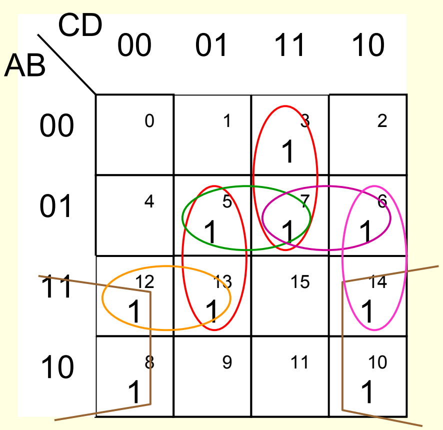


More examples

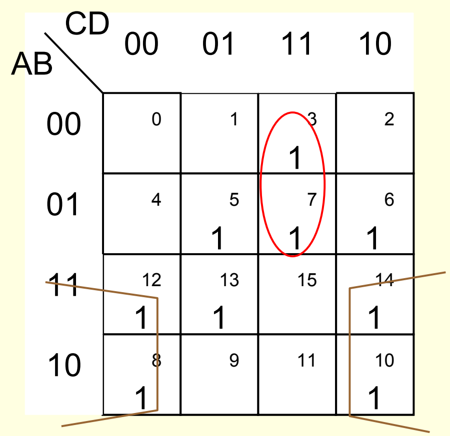
- Use a K-map to simplify the following Boolean function:

  $F(A,B,C,D) = \Pi M(0,1,2,4,9,11,15)$

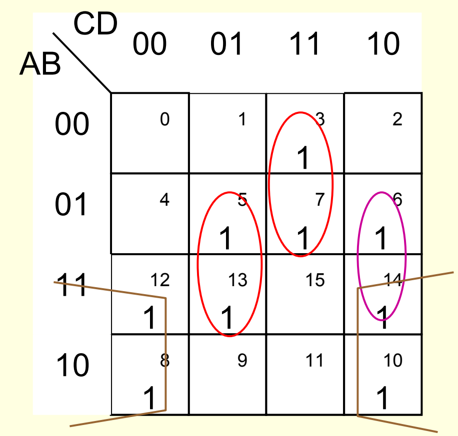
  $F(A,B,C,D) = \Sigma m(3,5,6,7,8,10,12,13,14)$



Prime Implicants



Essential Prime Implicants



Final Expression

$F(A,B,C,D) = BC'D + AD' + A'CD + BCD'$

“Don’t Care” On Karnaugh Map

- “Don’t Care” conditions
  - Eg: in BCD code which have 6 invalid combination: 1010, 1011, 1100, 1101, 1110, and 1111.
  - A “X” is placed in the cell.
  - It can be treat as either 1 or 0.
  - When grouping, the Xs, it can be treated as 1s to make a larger grouping.

The K-Map POS Minimization

- Determine the minimum SOP expression
  - Determine the binary value of each sum term in the standard POS expression.
  - As each sum term is evaluated, place a 0 on the Karnaugh map in the corresponding cell.

The K-Map POS Minimization

- Ex: Map the following standard POS expression on a K-map.

1) $$\left(A+B+C\right)\left(A+B+\overline{C}\right)\left(A+\overline{B}+C\right)\left(A+\overline{B}+\overline{C}\right)\left(\overline{A}+\overline{B}+C\right)$$

2) $$\left(\overline{A}+\overline{B}+C+D\right)\left(\overline{A}+B+\overline{C}+\overline{D}\right)\left(A+B+\overline{C}+D\right)\left(\overline{A}+\overline{B}+\overline{C}+\overline{D}\right)\left(A+B+\overline{C}+\overline{D}\right)$$